Project 5

Ayush Jain

29/01/2020

# Deploying necessary Libraries to the Code.  
  
library(corrplot)

## corrplot 0.84 loaded

library(DataExplorer)  
library(ggplot2)  
library(car)

## Loading required package: carData

library(caret)

## Loading required package: lattice

library(caTools)  
library(psych)

##   
## Attaching package: 'psych'

## The following object is masked from 'package:car':  
##   
## logit

## The following objects are masked from 'package:ggplot2':  
##   
## %+%, alpha

library(ggbiplot)

## Loading required package: plyr

## Loading required package: scales

##   
## Attaching package: 'scales'

## The following objects are masked from 'package:psych':  
##   
## alpha, rescale

## Loading required package: grid

library(ipred)  
library(e1071)  
library(rpart)  
library(DMwR)

## Warning: package 'DMwR' was built under R version 3.6.2

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

##   
## Attaching package: 'DMwR'

## The following object is masked from 'package:plyr':  
##   
## join

library(xgboost)

## Warning: package 'xgboost' was built under R version 3.6.2

library(blorr)

## Warning: package 'blorr' was built under R version 3.6.2

library(lmtest)

## Warning: package 'lmtest' was built under R version 3.6.2

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 3.6.2

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

library(pscl)

## Warning: package 'pscl' was built under R version 3.6.2

## Classes and Methods for R developed in the  
## Political Science Computational Laboratory  
## Department of Political Science  
## Stanford University  
## Simon Jackman  
## hurdle and zeroinfl functions by Achim Zeileis

# Setting the Working Directory.  
  
setwd("D:/Great Learning/Machine Learning/Project 5")  
  
# Reading the dataset.  
  
cars<- read.csv("Cars\_edited.csv")  
  
str(cars)

## 'data.frame': 444 obs. of 9 variables:  
## $ Age : int 28 23 29 28 27 26 28 26 22 27 ...  
## $ Gender : Factor w/ 2 levels "Female","Male": 2 1 2 1 2 2 2 1 2 2 ...  
## $ Engineer : int 0 1 1 1 1 1 1 1 1 1 ...  
## $ MBA : int 0 0 0 1 0 0 0 0 0 0 ...  
## $ Work.Exp : int 4 4 7 5 4 4 5 3 1 4 ...  
## $ Salary : num 14.3 8.3 13.4 13.4 13.4 12.3 14.4 10.5 7.5 13.5 ...  
## $ Distance : num 3.2 3.3 4.1 4.5 4.6 4.8 5.1 5.1 5.1 5.2 ...  
## $ license : int 0 0 0 0 0 1 0 0 0 0 ...  
## $ Transport: Factor w/ 3 levels "2Wheeler","Car",..: 3 3 3 3 3 3 1 3 3 3 ...

summary(cars)

## Age Gender Engineer MBA   
## Min. :18.00 Female:128 Min. :0.0000 Min. :0.0000   
## 1st Qu.:25.00 Male :316 1st Qu.:1.0000 1st Qu.:0.0000   
## Median :27.00 Median :1.0000 Median :0.0000   
## Mean :27.75 Mean :0.7545 Mean :0.2528   
## 3rd Qu.:30.00 3rd Qu.:1.0000 3rd Qu.:1.0000   
## Max. :43.00 Max. :1.0000 Max. :1.0000   
## NA's :1   
## Work.Exp Salary Distance license   
## Min. : 0.0 Min. : 6.50 Min. : 3.20 Min. :0.0000   
## 1st Qu.: 3.0 1st Qu.: 9.80 1st Qu.: 8.80 1st Qu.:0.0000   
## Median : 5.0 Median :13.60 Median :11.00 Median :0.0000   
## Mean : 6.3 Mean :16.24 Mean :11.32 Mean :0.2342   
## 3rd Qu.: 8.0 3rd Qu.:15.72 3rd Qu.:13.43 3rd Qu.:0.0000   
## Max. :24.0 Max. :57.00 Max. :23.40 Max. :1.0000   
##   
## Transport   
## 2Wheeler : 83   
## Car : 61   
## Public Transport:300   
##   
##   
##   
##

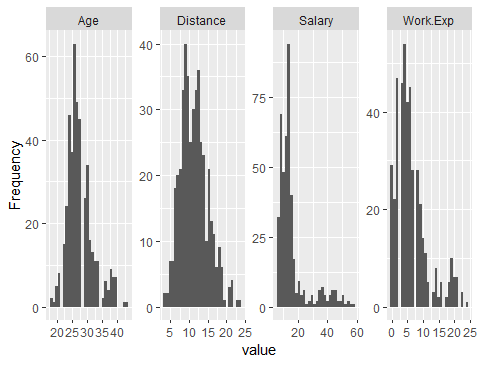
dim(cars)

## [1] 444 9

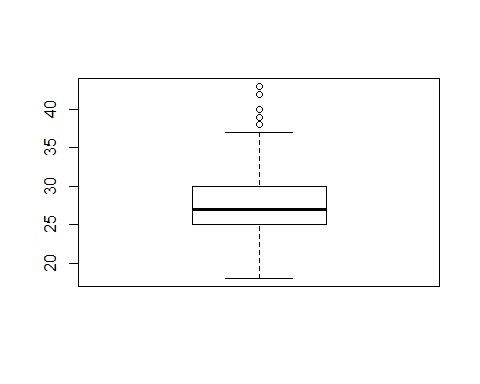
cars<- na.omit(cars)  
names(cars)

## [1] "Age" "Gender" "Engineer" "MBA" "Work.Exp" "Salary"   
## [7] "Distance" "license" "Transport"

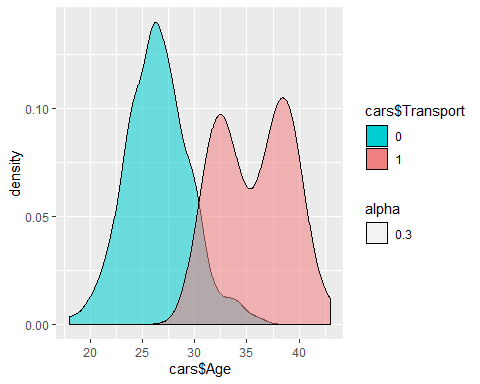
# Converting Dependent variables to 0 and 1  
cars$Transport <- ifelse(cars$Transport == "Car",1,0)  
cars$Transport <- as.factor(cars$Transport)  
  
  
plot\_histogram(cars)



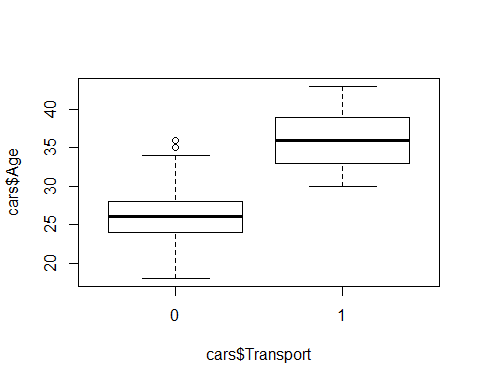
boxplot(cars$Age)



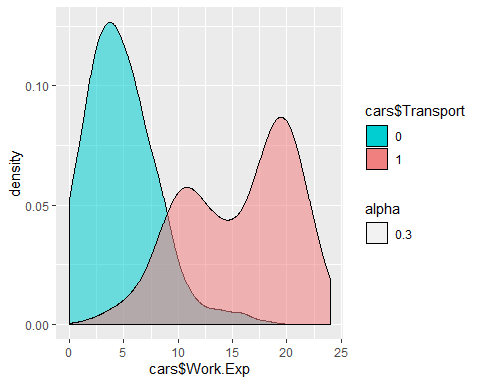
ggplot(cars, aes(x=cars$Age)) +  
 geom\_density(aes(fill =cars$Transport, alpha = 0.3)) +  
 scale\_color\_manual(values = c("#868686FF", "#EFC000FF")) +   
 scale\_fill\_manual(values = c("darkturquoise","lightcoral","lightgreen"))



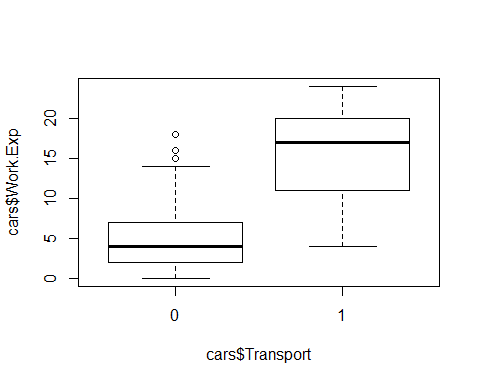
boxplot(cars$Age~ cars$Transport)



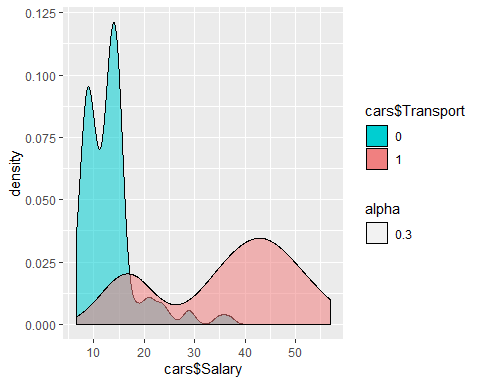
ggplot(cars, aes(x=cars$Work.Exp)) +  
 geom\_density(aes(fill =cars$Transport, alpha = 0.3)) +  
 scale\_color\_manual(values = c("#868686FF", "#EFC000FF")) +   
 scale\_fill\_manual(values = c("darkturquoise","lightcoral","lightgreen"))



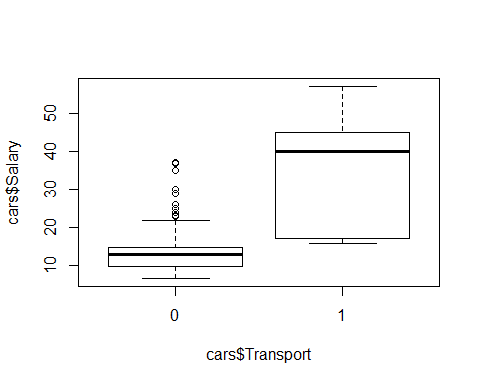
boxplot(cars$Work.Exp~ cars$Transport)



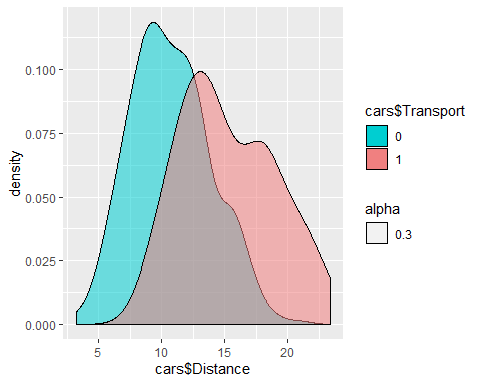
ggplot(cars, aes(x=cars$Salary)) +  
 geom\_density(aes(fill =cars$Transport, alpha = 0.3)) +  
 scale\_color\_manual(values = c("#868686FF", "#EFC000FF")) +   
 scale\_fill\_manual(values = c("darkturquoise","lightcoral","lightgreen"))



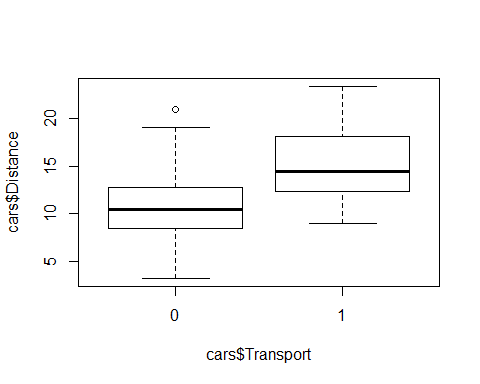
boxplot(cars$Salary~ cars$Transport)



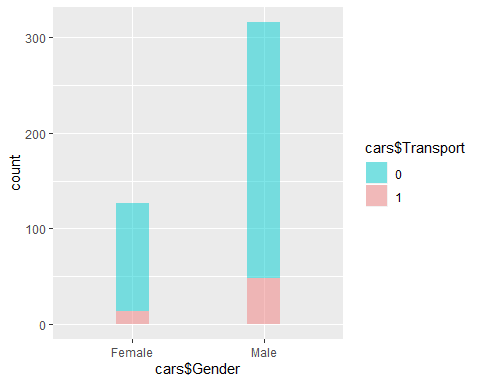
ggplot(cars, aes(x=cars$Distance)) +  
 geom\_density(aes(fill =cars$Transport, alpha = 0.3)) +  
 scale\_color\_manual(values = c("#868686FF", "#EFC000FF")) +   
 scale\_fill\_manual(values = c("darkturquoise","lightcoral","lightgreen"))



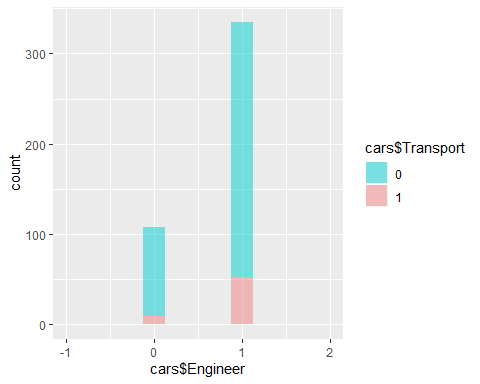
boxplot(cars$Distance~ cars$Transport)



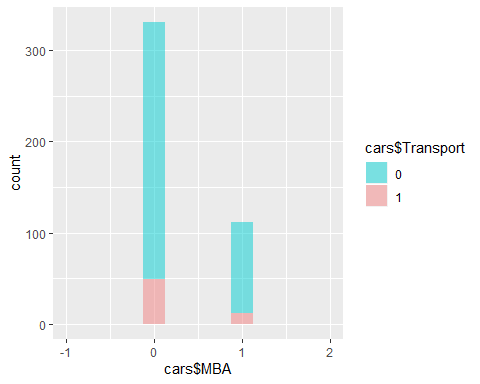
ggplot(cars,aes(x = cars$Gender, fill = cars$Transport)) +  
 geom\_bar(width = 0.25,alpha = 0.5) +   
 scale\_fill\_manual(values = c("darkturquoise","lightcoral","lightgreen"))



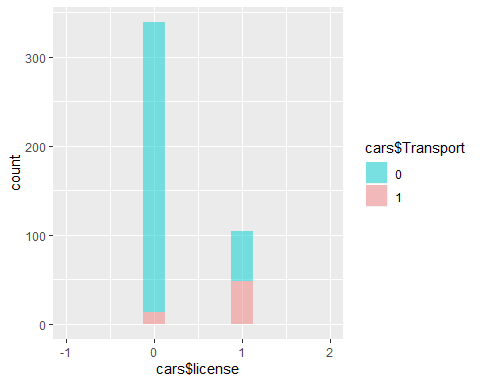
ggplot(cars,aes(x = cars$Engineer, fill = cars$Transport)) +  
 geom\_bar(width = 0.25,alpha = 0.5) +   
 scale\_fill\_manual(values = c("darkturquoise","lightcoral","lightgreen")) + xlim(-1,2)



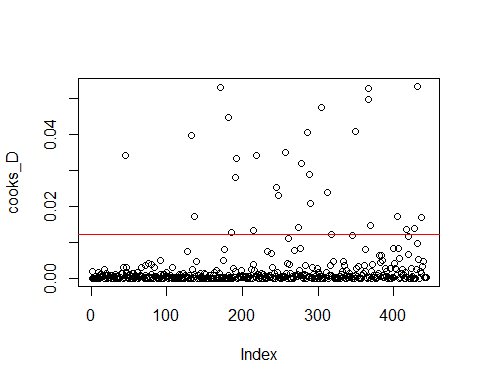
ggplot(cars,aes(x = cars$MBA, fill = cars$Transport)) +  
 geom\_bar(width = 0.25,alpha = 0.5) +   
 scale\_fill\_manual(values = c("darkturquoise","lightcoral","lightgreen")) + xlim(-1,2)



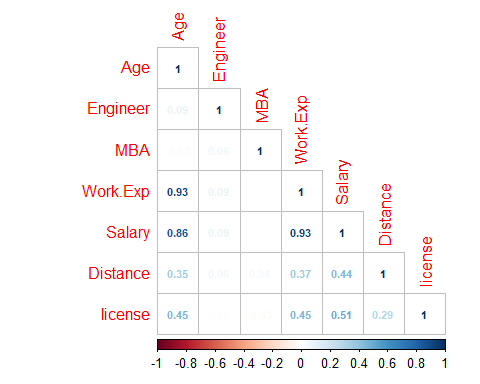
ggplot(cars,aes(x = cars$license, fill = cars$Transport)) +  
 geom\_bar(width = 0.25,alpha = 0.5) +   
 scale\_fill\_manual(values = c("darkturquoise","lightcoral","lightgreen")) + xlim(-1,2)



# Performing Cooks Distance  
  
cd\_lm <- lm(as.numeric(cars$Transport)~.,data = cars)  
cooks\_D <- cooks.distance(cd\_lm)  
  
plot(cooks\_D)  
abline(h=4\*mean(cooks\_D,na.rm = TRUE),col="red")



# Performing Correlation Matrix and Plotting it.  
  
mat<- cor(cars[,-c(2,9)])  
corrplot(mat,method = "number", type = "lower", number.cex = .70)



prop.table(table(cars$Transport))

##   
## 0 1   
## 0.8623025 0.1376975

#Splitting the data into Train and Test with a Ratio of 70 and 30 resp.  
str(cars)

## 'data.frame': 443 obs. of 9 variables:  
## $ Age : int 28 23 29 28 27 26 28 26 22 27 ...  
## $ Gender : Factor w/ 2 levels "Female","Male": 2 1 2 1 2 2 2 1 2 2 ...  
## $ Engineer : int 0 1 1 1 1 1 1 1 1 1 ...  
## $ MBA : int 0 0 0 1 0 0 0 0 0 0 ...  
## $ Work.Exp : int 4 4 7 5 4 4 5 3 1 4 ...  
## $ Salary : num 14.3 8.3 13.4 13.4 13.4 12.3 14.4 10.5 7.5 13.5 ...  
## $ Distance : num 3.2 3.3 4.1 4.5 4.6 4.8 5.1 5.1 5.1 5.2 ...  
## $ license : int 0 0 0 0 0 1 0 0 0 0 ...  
## $ Transport: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...  
## - attr(\*, "na.action")= 'omit' Named int 145  
## ..- attr(\*, "names")= chr "145"

set.seed(123)  
index <- sample.split(cars, SplitRatio = .70)  
trainData <- subset(cars,index==TRUE)  
dim(trainData)

## [1] 296 9

testData <- subset(cars,index==FALSE)  
dim(testData)

## [1] 147 9

balanced\_trainData <- SMOTE(Transport~., trainData, perc.over = 325,k = 5,perc.under =134)  
str(balanced\_trainData)

## 'data.frame': 344 obs. of 9 variables:  
## $ Age : num 26 24 29 23 30 26 27 33 22 30 ...  
## $ Gender : Factor w/ 2 levels "Female","Male": 1 2 1 1 1 2 2 2 2 2 ...  
## $ Engineer : num 1 0 0 1 0 1 1 1 1 0 ...  
## $ MBA : num 0 1 1 1 0 1 0 1 0 0 ...  
## $ Work.Exp : num 4 2 7 4 6 4 7 11 0 8 ...  
## $ Salary : num 12.9 8.9 14.6 8.4 15.6 12.9 16.6 15.6 6.9 14.6 ...  
## $ Distance : num 11.1 13.4 10.9 7.1 11.6 10 6.4 9.3 13.2 6.1 ...  
## $ license : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Transport: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...

prop.table(table(balanced\_trainData$Transport))

##   
## 0 1   
## 0.5 0.5

# Scaling the data.  
  
scale\_train <- scale(balanced\_trainData[,-c(2,3,4,8,9)])  
scale\_train <- cbind(balanced\_trainData[,c(9,2,3,4,8)], scale\_train)  
str(scale\_train)

## 'data.frame': 344 obs. of 9 variables:  
## $ Transport: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Gender : Factor w/ 2 levels "Female","Male": 1 2 1 1 1 2 2 2 2 2 ...  
## $ Engineer : num 1 0 0 1 0 1 1 1 1 0 ...  
## $ MBA : num 0 1 1 1 0 1 0 1 0 0 ...  
## $ license : num 0 0 0 0 0 0 0 0 0 0 ...  
## $ Age : num -0.92 -1.292 -0.361 -1.478 -0.175 ...  
## $ Work.Exp : num -0.942 -1.253 -0.475 -0.942 -0.631 ...  
## $ Salary : num -0.79 -1.058 -0.676 -1.091 -0.608 ...  
## $ Distance : num -0.429 0.167 -0.481 -1.465 -0.299 ...

scale\_test <- scale(testData[,-c(2,3,4,8,9)])  
scale\_test <- cbind(testData[,c(9,2,3,4,8)], scale\_test)  
str(scale\_test)

## 'data.frame': 147 obs. of 9 variables:  
## $ Transport: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Gender : Factor w/ 2 levels "Female","Male": 1 2 1 2 2 2 2 1 1 1 ...  
## $ Engineer : int 1 1 1 1 1 1 1 1 0 0 ...  
## $ MBA : int 1 0 0 0 0 0 0 0 0 0 ...  
## $ license : int 0 0 0 0 1 0 0 0 0 0 ...  
## $ Age : num 0.0562 -0.1798 -0.4159 -0.8879 -0.1798 ...  
## $ Work.Exp : num -0.248 -0.45 -0.651 -0.853 -0.45 ...  
## $ Salary : num -0.269 -0.269 -0.575 -0.787 -0.269 ...  
## $ Distance : num -1.92 -1.89 -1.75 -1.67 -1.64 ...

#Performing Logistic Model on Train data.  
  
logit <- glm(scale\_train$Transport~., data = scale\_train, family = "binomial")

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

summary(logit)

##   
## Call:  
## glm(formula = scale\_train$Transport ~ ., family = "binomial",   
## data = scale\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -3.3970 -0.0002 0.0000 0.0001 1.3550   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -3.6126 1.6958 -2.130 0.033151 \*   
## GenderMale -0.1713 0.8499 -0.202 0.840228   
## Engineer 1.4930 1.1247 1.327 0.184353   
## MBA 2.2322 1.4762 1.512 0.130500   
## license 3.1087 1.2313 2.525 0.011581 \*   
## Age 26.1337 7.4271 3.519 0.000434 \*\*\*  
## Work.Exp -17.0228 5.1573 -3.301 0.000964 \*\*\*  
## Salary 7.5424 2.6182 2.881 0.003967 \*\*   
## Distance 4.8639 1.6831 2.890 0.003855 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 476.885 on 343 degrees of freedom  
## Residual deviance: 44.474 on 335 degrees of freedom  
## AIC: 62.474  
##   
## Number of Fisher Scoring iterations: 11

vif(logit)

## Gender Engineer MBA license Age Work.Exp Salary   
## 1.255118 1.291604 2.543172 1.526840 29.830045 46.429945 8.551596   
## Distance   
## 9.821118

logit\_refined <- glm(scale\_train$Transport~scale\_train$Age+scale\_train$Gender+scale\_train$Engineer+scale\_train$MBA+scale\_train$Salary+scale\_train$Distance+scale\_train$license,  
 data = scale\_train, family = "binomial")  
summary(logit)

##   
## Call:  
## glm(formula = scale\_train$Transport ~ ., family = "binomial",   
## data = scale\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -3.3970 -0.0002 0.0000 0.0001 1.3550   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -3.6126 1.6958 -2.130 0.033151 \*   
## GenderMale -0.1713 0.8499 -0.202 0.840228   
## Engineer 1.4930 1.1247 1.327 0.184353   
## MBA 2.2322 1.4762 1.512 0.130500   
## license 3.1087 1.2313 2.525 0.011581 \*   
## Age 26.1337 7.4271 3.519 0.000434 \*\*\*  
## Work.Exp -17.0228 5.1573 -3.301 0.000964 \*\*\*  
## Salary 7.5424 2.6182 2.881 0.003967 \*\*   
## Distance 4.8639 1.6831 2.890 0.003855 \*\*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 476.885 on 343 degrees of freedom  
## Residual deviance: 44.474 on 335 degrees of freedom  
## AIC: 62.474  
##   
## Number of Fisher Scoring iterations: 11

vif(logit\_refined)

## scale\_train$Age scale\_train$Gender scale\_train$Engineer   
## 1.405195 1.113576 1.221754   
## scale\_train$MBA scale\_train$Salary scale\_train$Distance   
## 1.147239 1.501177 1.336347   
## scale\_train$license   
## 1.360300

# Calculating Likelihood Test  
?lrtest

## starting httpd help server ...

## done

logit\_likelihood <- lrtest(logit\_refined)  
logit\_likelihood

## Likelihood ratio test  
##   
## Model 1: scale\_train$Transport ~ scale\_train$Age + scale\_train$Gender +   
## scale\_train$Engineer + scale\_train$MBA + scale\_train$Salary +   
## scale\_train$Distance + scale\_train$license  
## Model 2: scale\_train$Transport ~ 1  
## #Df LogLik Df Chisq Pr(>Chisq)   
## 1 8 -38.34   
## 2 1 -238.44 -7 400.21 < 2.2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

# Calculating Psudo R Sq  
  
logit\_rsq <- pR2(logit\_refined)  
logit\_rsq

## llh llhNull G2 McFadden r2ML   
## -38.3398975 -238.4426301 400.2054653 0.8392070 0.6875741   
## r2CU   
## 0.9167655

# Calculating Odds Ratio  
  
logit\_odds <- exp(logit\_refined$coefficients)  
print(logit\_odds, digits = 20)

## (Intercept) scale\_train$Age scale\_train$GenderMale   
## 0.24056289445230594 311.93495401982949033 0.74928475762118107   
## scale\_train$Engineer scale\_train$MBA scale\_train$Salary   
## 4.39181032734932764 0.99719695708375100 1.39246074991714752   
## scale\_train$Distance scale\_train$license   
## 3.36071669068749301 7.00844627690320809

# Predicting values on Train Data.  
  
pred\_logit\_train <- predict(logit\_refined, scale\_train, type = "response")  
  
pred\_logit\_train\_class <- ifelse(pred\_logit\_train <.5,0,1)  
head(pred\_logit\_train\_class)

## 228 331 218 46 247 178   
## 0 0 0 0 0 0

prop.table(table(pred\_logit\_train\_class))

## pred\_logit\_train\_class  
## 0 1   
## 0.4912791 0.5087209

# Calculating Baseline   
prop.table(table(cars$Transport))

##   
## 0 1   
## 0.8623025 0.1376975

# Creating Performance Matrix on Train  
  
pred\_logit\_train\_class<- as.factor(pred\_logit\_train\_class)  
caret::confusionMatrix(pred\_logit\_train\_class,scale\_train$Transport)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 164 5  
## 1 8 167  
##   
## Accuracy : 0.9622   
## 95% CI : (0.9362, 0.9797)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9244   
##   
## Mcnemar's Test P-Value : 0.5791   
##   
## Sensitivity : 0.9535   
## Specificity : 0.9709   
## Pos Pred Value : 0.9704   
## Neg Pred Value : 0.9543   
## Prevalence : 0.5000   
## Detection Rate : 0.4767   
## Detection Prevalence : 0.4913   
## Balanced Accuracy : 0.9622   
##   
## 'Positive' Class : 0   
##

# Performing Prediction on Test Data  
  
pred\_logit\_test <- predict(logit, scale\_test, type = "response")  
pred\_logit\_test\_class <- ifelse(pred\_logit\_test <.5,0,1)  
head(pred\_logit\_test\_class)

## 4 5 8 13 14 17   
## 0 0 0 0 0 1

# Creating performance Matrix on Test  
length(pred\_logit\_test\_class)

## [1] 147

pred\_logit\_test\_class <- as.factor(pred\_logit\_test\_class)  
caret::confusionMatrix(pred\_logit\_test\_class,scale\_test$Transport)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 83 0  
## 1 46 18  
##   
## Accuracy : 0.6871   
## 95% CI : (0.6055, 0.7609)  
## No Information Rate : 0.8776   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.3065   
##   
## Mcnemar's Test P-Value : 3.247e-11   
##   
## Sensitivity : 0.6434   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.2813   
## Prevalence : 0.8776   
## Detection Rate : 0.5646   
## Detection Prevalence : 0.5646   
## Balanced Accuracy : 0.8217   
##   
## 'Positive' Class : 0   
##

# Performing K- Nearest Neighbor   
ctrl <- trainControl(method = "cv", number = 3)  
knnModel <- train(Transport~.,data = scale\_train,method = "knn",  
 trControl = ctrl,  
 tuneLength = 10)  
knnModel$bestTune

## k  
## 7 17

summary(knnModel)

## Length Class Mode   
## learn 2 -none- list   
## k 1 -none- numeric   
## theDots 0 -none- list   
## xNames 8 -none- character  
## problemType 1 -none- character  
## tuneValue 1 data.frame list   
## obsLevels 2 -none- character  
## param 0 -none- list

# Predicting Value on Train data   
  
pred\_knn\_Train <- predict(knnModel,scale\_train)  
  
# Creating Perfromance Matrix  
  
caret::confusionMatrix(pred\_knn\_Train,scale\_train$Transport)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 164 5  
## 1 8 167  
##   
## Accuracy : 0.9622   
## 95% CI : (0.9362, 0.9797)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9244   
##   
## Mcnemar's Test P-Value : 0.5791   
##   
## Sensitivity : 0.9535   
## Specificity : 0.9709   
## Pos Pred Value : 0.9704   
## Neg Pred Value : 0.9543   
## Prevalence : 0.5000   
## Detection Rate : 0.4767   
## Detection Prevalence : 0.4913   
## Balanced Accuracy : 0.9622   
##   
## 'Positive' Class : 0   
##

# Predicting Values on Test data.  
  
pred\_knn\_test <- predict(knnModel, scale\_test)  
  
caret::confusionMatrix(pred\_knn\_test, scale\_test$Transport)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 87 0  
## 1 42 18  
##   
## Accuracy : 0.7143   
## 95% CI : (0.634, 0.7857)  
## No Information Rate : 0.8776   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.3366   
##   
## Mcnemar's Test P-Value : 2.509e-10   
##   
## Sensitivity : 0.6744   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.3000   
## Prevalence : 0.8776   
## Detection Rate : 0.5918   
## Detection Prevalence : 0.5918   
## Balanced Accuracy : 0.8372   
##   
## 'Positive' Class : 0   
##

# Performing Naive Bayes Model  
  
NBModel <- naiveBayes(Transport~., data = scale\_train)  
summary(NBModel)

## Length Class Mode   
## apriori 2 table numeric   
## tables 8 -none- list   
## levels 2 -none- character  
## isnumeric 8 -none- logical   
## call 4 -none- call

# Prediction on Train data.  
  
pred\_nb\_train <- predict(NBModel, scale\_train)  
  
  
# Creating Performance Matrix on Traindata   
caret::confusionMatrix(pred\_nb\_train, scale\_train$Transport)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 167 18  
## 1 5 154  
##   
## Accuracy : 0.9331   
## 95% CI : (0.9014, 0.9571)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : < 2e-16   
##   
## Kappa : 0.8663   
##   
## Mcnemar's Test P-Value : 0.01234   
##   
## Sensitivity : 0.9709   
## Specificity : 0.8953   
## Pos Pred Value : 0.9027   
## Neg Pred Value : 0.9686   
## Prevalence : 0.5000   
## Detection Rate : 0.4855   
## Detection Prevalence : 0.5378   
## Balanced Accuracy : 0.9331   
##   
## 'Positive' Class : 0   
##

# Prediction on Test data.  
  
pred\_nb\_test <- predict(NBModel, scale\_test)  
  
# Creating Performance Matrix on Test Data   
caret::confusionMatrix(pred\_nb\_test, scale\_test$Transport)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 90 0  
## 1 39 18  
##   
## Accuracy : 0.7347   
## 95% CI : (0.6556, 0.804)  
## No Information Rate : 0.8776   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.3611   
##   
## Mcnemar's Test P-Value : 1.166e-09   
##   
## Sensitivity : 0.6977   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.3158   
## Prevalence : 0.8776   
## Detection Rate : 0.6122   
## Detection Prevalence : 0.6122   
## Balanced Accuracy : 0.8488   
##   
## 'Positive' Class : 0   
##

# Performing Bagging Model  
  
bagModel <- bagging(as.numeric(Transport)~.,data = scale\_train,  
 control = rpart.control(maxdepth = 5, minsplit = 3))  
?bagging  
summary(bagModel)

## Length Class Mode   
## y 344 -none- numeric  
## X 8 data.frame list   
## mtrees 25 -none- list   
## OOB 1 -none- logical  
## comb 1 -none- logical  
## call 4 -none- call

#Predicting Model on Train data  
  
pred\_bag\_train <- predict(bagModel, data= scale\_train)  
pred\_bag\_train1 <- ifelse(pred\_bag\_train<0.5,0,1)  
pred\_bag\_train1 <- as.factor(pred\_bag\_train1)  
  
caret::confusionMatrix(pred\_bag\_train1,scale\_train$Transport)

## Warning in confusionMatrix.default(pred\_bag\_train1, scale\_train$Transport):  
## Levels are not in the same order for reference and data. Refactoring data  
## to match.

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 0 0  
## 1 172 172  
##   
## Accuracy : 0.5   
## 95% CI : (0.4459, 0.5541)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : 0.5215   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.0   
## Specificity : 1.0   
## Pos Pred Value : NaN   
## Neg Pred Value : 0.5   
## Prevalence : 0.5   
## Detection Rate : 0.0   
## Detection Prevalence : 0.0   
## Balanced Accuracy : 0.5   
##   
## 'Positive' Class : 0   
##

# Predicting Model on Test data  
  
pred\_bag\_test <- predict(bagModel, newdata = scale\_test)  
pred\_bag\_test1 <- ifelse(pred\_bag\_test<0.5,0,1)  
pred\_bag\_test1 <- as.factor(pred\_bag\_test1)  
  
caret::confusionMatrix(pred\_bag\_test1,scale\_test$Transport)

## Warning in confusionMatrix.default(pred\_bag\_test1, scale\_test$Transport):  
## Levels are not in the same order for reference and data. Refactoring data  
## to match.

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 0 0  
## 1 129 18  
##   
## Accuracy : 0.1224   
## 95% CI : (0.0742, 0.1866)  
## No Information Rate : 0.8776   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.0000   
## Specificity : 1.0000   
## Pos Pred Value : NaN   
## Neg Pred Value : 0.1224   
## Prevalence : 0.8776   
## Detection Rate : 0.0000   
## Detection Prevalence : 0.0000   
## Balanced Accuracy : 0.5000   
##   
## 'Positive' Class : 0   
##

# Performing boosting Model  
  
str(scale\_test)

## 'data.frame': 147 obs. of 9 variables:  
## $ Transport: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Gender : Factor w/ 2 levels "Female","Male": 1 2 1 2 2 2 2 1 1 1 ...  
## $ Engineer : int 1 1 1 1 1 1 1 1 0 0 ...  
## $ MBA : int 1 0 0 0 0 0 0 0 0 0 ...  
## $ license : int 0 0 0 0 1 0 0 0 0 0 ...  
## $ Age : num 0.0562 -0.1798 -0.4159 -0.8879 -0.1798 ...  
## $ Work.Exp : num -0.248 -0.45 -0.651 -0.853 -0.45 ...  
## $ Salary : num -0.269 -0.269 -0.575 -0.787 -0.269 ...  
## $ Distance : num -1.92 -1.89 -1.75 -1.67 -1.64 ...

scale\_train$Gender <- as.numeric(scale\_train$Gender)  
scale\_test$Gender <- as.numeric(scale\_test$Gender)  
features\_train <- as.matrix(scale\_train[,2:9])  
#str(features\_train)  
label\_train <- as.matrix(scale\_train[,1])  
str(label\_train)

## chr [1:344, 1] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" ...

features\_test <- as.matrix(scale\_test[,2:9])  
#str(features\_test)  
  
xgbModel <- xgboost(  
 data = features\_train,  
 label = label\_train,  
 eta = 1,  
 max\_depth = 100,  
 min\_child\_weight = 3,  
 nrounds = 1000,  
 nfold = 10,  
 objective = "binary:logistic",  
 verbose = 0,  
 early\_stopping\_rounds = 10)  
  
summary(xgbModel)

## Length Class Mode   
## handle 1 xgb.Booster.handle externalptr  
## raw 5301 -none- raw   
## best\_iteration 1 -none- numeric   
## best\_ntreelimit 1 -none- numeric   
## best\_score 1 -none- numeric   
## niter 1 -none- numeric   
## evaluation\_log 2 data.table list   
## call 18 -none- call   
## params 6 -none- list   
## callbacks 2 -none- list   
## feature\_names 8 -none- character   
## nfeatures 1 -none- numeric

?xgboost  
# Performing Prediction on Train data.  
str(label\_train)

## chr [1:344, 1] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" ...

pred\_xgb\_train <- predict(xgbModel, newdata = features\_train)  
pred\_xgb\_train1 <- ifelse(pred\_xgb\_train<.5,0,1)  
pred\_xgb\_train1<- as.factor(pred\_xgb\_train1)  
caret::confusionMatrix(pred\_xgb\_train1, scale\_train$Transport)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 169 1  
## 1 3 171  
##   
## Accuracy : 0.9884   
## 95% CI : (0.9705, 0.9968)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9767   
##   
## Mcnemar's Test P-Value : 0.6171   
##   
## Sensitivity : 0.9826   
## Specificity : 0.9942   
## Pos Pred Value : 0.9941   
## Neg Pred Value : 0.9828   
## Prevalence : 0.5000   
## Detection Rate : 0.4913   
## Detection Prevalence : 0.4942   
## Balanced Accuracy : 0.9884   
##   
## 'Positive' Class : 0   
##

# Performing Prediction on Test data.  
str(label\_train)

## chr [1:344, 1] "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" "0" ...

pred\_xgb\_test <- predict(xgbModel, newdata= features\_test)  
pred\_xgb\_test1 <- ifelse(pred\_xgb\_test<.5,0,1)  
pred\_xgb\_test1<- as.factor(pred\_xgb\_test1)  
caret::confusionMatrix(pred\_xgb\_test1, scale\_test$Transport)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 80 0  
## 1 49 18  
##   
## Accuracy : 0.6667   
## 95% CI : (0.5843, 0.7422)  
## No Information Rate : 0.8776   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.2856   
##   
## Mcnemar's Test P-Value : 7.025e-12   
##   
## Sensitivity : 0.6202   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.2687   
## Prevalence : 0.8776   
## Detection Rate : 0.5442   
## Detection Prevalence : 0.5442   
## Balanced Accuracy : 0.8101   
##   
## 'Positive' Class : 0   
##

vec\_xgb <- vector()  
lr <- c(0.001,0.01,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1)  
md <- c(1,3,5,7,9,11,13,15)  
nr <- c(2,25,50,75,100,500,1000)  
  
for (i in nr){  
 xgbModel\_ref <- xgboost(  
 data = features\_train,  
 label = label\_train,  
 eta = 1,  
 max\_depth = 100,  
 min\_child\_weight = 3,  
 nrounds = i,  
 nfold = 10,  
 objective = "binary:logistic",  
 verbose = 0,  
 early\_stopping\_rounds = 10)  
   
 xgb.pred.class <- predict(xgbModel\_ref,features\_test)  
 vec\_xgb <- cbind(vec\_xgb,sum(scale\_test$Transport==1 & xgb.pred.class > 0.5))  
}  
vec\_xgb

## [,1] [,2] [,3] [,4] [,5] [,6] [,7]  
## [1,] 18 18 18 18 18 18 18

xgbModel1 <- xgboost(  
 data = features\_train,  
 label = label\_train,  
 eta = 1,  
 max\_depth = 1,  
 min\_child\_weight = 3,  
 nrounds = 2,  
 nfold = 10,  
 objective = "binary:logistic",  
 verbose = 0,  
 early\_stopping\_rounds = 10)  
  
  
# Performing Prediction on Train data.  
pred\_xgb\_ref\_train <- predict(xgbModel1, newdata = features\_train)  
pred\_xgb\_ref\_train1 <- ifelse(pred\_xgb\_ref\_train<.5,0,1)  
pred\_xgb\_ref\_train1<- as.factor(pred\_xgb\_ref\_train1)  
caret::confusionMatrix(pred\_xgb\_train1, scale\_train$Transport)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 169 1  
## 1 3 171  
##   
## Accuracy : 0.9884   
## 95% CI : (0.9705, 0.9968)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : <2e-16   
##   
## Kappa : 0.9767   
##   
## Mcnemar's Test P-Value : 0.6171   
##   
## Sensitivity : 0.9826   
## Specificity : 0.9942   
## Pos Pred Value : 0.9941   
## Neg Pred Value : 0.9828   
## Prevalence : 0.5000   
## Detection Rate : 0.4913   
## Detection Prevalence : 0.4942   
## Balanced Accuracy : 0.9884   
##   
## 'Positive' Class : 0   
##

# Performing Prediction on Test data.  
  
pred\_xgb\_ref\_test <- predict(xgbModel1, newdata= features\_test)  
pred\_xgb\_ref\_test1 <- ifelse(pred\_xgb\_ref\_test<.5,0,1)  
pred\_xgb\_ref\_test1<- as.factor(pred\_xgb\_ref\_test1)  
caret::confusionMatrix(pred\_xgb\_ref\_test1, scale\_test$Transport)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 79 0  
## 1 50 18  
##   
## Accuracy : 0.6599   
## 95% CI : (0.5772, 0.7359)  
## No Information Rate : 0.8776   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.279   
##   
## Mcnemar's Test P-Value : 4.219e-12   
##   
## Sensitivity : 0.6124   
## Specificity : 1.0000   
## Pos Pred Value : 1.0000   
## Neg Pred Value : 0.2647   
## Prevalence : 0.8776   
## Detection Rate : 0.5374   
## Detection Prevalence : 0.5374   
## Balanced Accuracy : 0.8062   
##   
## 'Positive' Class : 0   
##